Ref. 3

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2003－249919
（P2003－249919A）
(43)公開日　平成15年９月５日(2003.9.5)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｌ　5/16		Ｈ０４Ｌ　5/16		５Ｋ０１８
29/08		13/00	３０７Ｂ	５Ｋ０３４

審査請求　有　請求項の数７　ＯＬ　（全 20 頁）

(21)出願番号　特願2002－350379(P2002－350379)
(22)出願日　平成14年12月２日(2002.12.2)
(31)優先権主張番号　特願2001－383324(P2001－383324)
(32)優先日　平成13年12月17日(2001.12.17)
(33)優先権主張国　日本（ＪＰ）

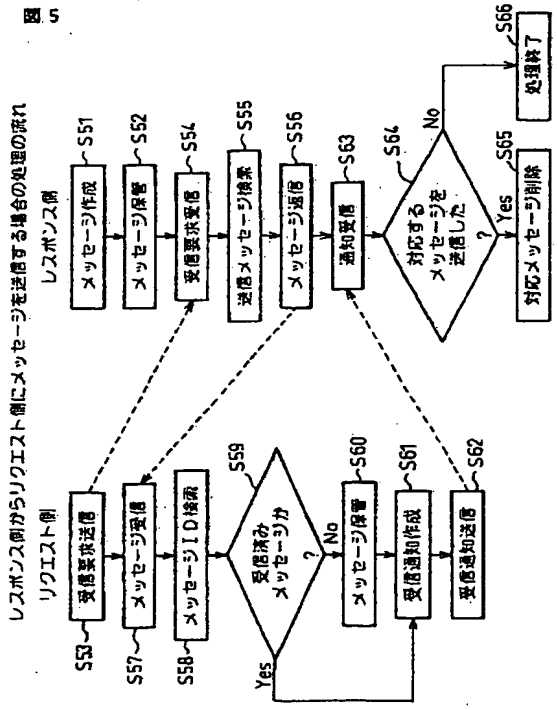
(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(72)発明者　野村　佳秀
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　原　裕貴
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(74)代理人　100077517
弁理士　石田　敬　（外４名）

最終頁に続く

(54)【発明の名称】　双方向通信方法

(57)【要約】
【課題】　片方向通信プロトコルで、双方向の送達確認を可能にし、送受信事実の否認を防止し、メッセージの一意性を保証する処理の自動化を容易にする。
【解決手段】　リクエスト側１１からの片方向のリクエスト／レスポンス型の同期通信のみを用いるシステムにおいて、レスポンス側１２からリクエスト側１１に向けてメッセージを送信する際は、リクエスト側がメッセージの受信要求を行ない、リクエスト側がメッセージを受信したことを、新たな通信でレスポンス側に受信通知するまで、レスポンス側は署名とＩＤを含む同じメッセージをリクエスト側に送信し続ける。

図５



1

【特許請求の範囲】

【請求項１】　リクエスト側からレスポンス側への片方向のリクエストと該リクエストに対する前記レスポンス側から前記リクエスト側へのレスポンスを行うリクエスト／レスポンス型の同期通信のみを用いるシステムにおいて、

前記リクエスト側から前記レスポンス側に向けてメッセージを送信する際は、前記レスポンス側がメッセージを受信したことを、前記リクエスト側に返信するまで、同じメッセージを前記レスポンス側に送信し続け、

前記レスポンス側から前記リクエスト側に向けてメッセージを送信する際は、前記リクエスト側がメッセージの受信要求を行ない、前記リクエスト側がメッセージを受信したことを、新たな通信で前記レスポンス側に受信通知するまで、前記レスポンス側は同じメッセージを前記リクエスト側に送信し続け、

それにより、双方向でメッセージの送受信を可能にした双方向通信手法。

【請求項２】　送信するメッセージ内に一意の識別子を付加し、受信側で重複チェックを行なうことにより、同じメッセージを重複して受信することを防ぐようにした請求項１記載の双方向通信手法。

【請求項３】　前記メッセージの形式にＸＭＬを利用し、前記メッセージとは異なる特定のネームスペースでメッセージ内に前記識別子を付加することで、メッセージの形式を変更せずに、同じメッセージを重複して受信することを防ぐようにした請求項２記載の双方向通信手法。

【請求項４】　前記メッセージの一意性は任意の形で実現し、メッセージの一意性を検証する際に、該メッセージのダイジェスト値を利用することで、同じメッセージを重複して受信することを防ぐようにした請求項１記載の双方向通信手法。

【請求項５】　受信メッセージをバッファに保管完了した後にメッセージを受信したことをメッセージ送信側に通知するようにし、前記バッファの容量不足により前記受信メッセージの保管に失敗した場合は前記バッファ内の古いメッセージを削除することにより受信メッセージを前記バッファに保管するようにした請求項１記載の双方向通信方法。

【請求項６】　前記メッセージの送信側が送信するメッセージに送信メッセージ用電子署名を付加し、該送信メッセージ用電子署名を受信側が保管することによって送信側による送信事実の否認を防止することができるようにすることと、

前記メッセージの受信側が前記メッセージの受信に応答して送信するメッセージ受信通知に、受信通知用電子署名を付加し、該受信通知用電子署名を、送信側が保管することによって受信側による受信事実の否認を防止することができるようにすることと、のすくなくとも一方を

2

採用した請求項２から４のいずれか一項記載の双方向通信手法。

【請求項７】　前記署名付きの受信メッセージを受信側のバッファに保管完了した後にメッセージを受信したことをメッセージ送信側に通知するようにし、前記バッファの容量不足により前記受信メッセージ及び前記署名の少なくとも一方の保管に失敗した場合は前記バッファ内の古いメッセージを削除した後に前記バッファの容量が依然として不足の場合は前記バッファ内の古い署名を削除することにより前記署名付きの受信メッセージを前記バッファに保管するようにした請求項６記載の双方向通信方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は双方向通信方法に関し、より詳細にはリクエスト側からレスポンス側への片方向のリクエストとそのリクエストに対するレスポンス側からリクエスト側へのレスポンスを行うリクエスト／レスポンス型の同期通信のみを用いるシステムにおいて、双方向でメッセージを送受信すること、信頼性を保証すること、及び送受信の否認を防止することを可能にした双方向通信手法に関する。

【０００２】

【従来の技術】図１はリクエスト／レスポンス型の同期通信のみを用いる従来のシステムの構成を示すブロック図である。図において、１１はリスエスト側であるクライアント、１２はレスポンス側であるサーバ、１３はクライアント１１の入口に設けられたファイアウオールである。このように、ファイアウオール１３が設けられていると、その内部とその外側との通信は、片方向のリクエスト／レスポンス型の同期通信を用いることが多い。

【０００３】

【発明が解決しようとする課題】片方向のリクエスト／レスポンス型の同期通信のみを用いた場合、従来のＨＴＴＰなどのプロトコルでは、図１に示すようにメッセージ送達とその確認はリクエスト側１１からレスポンス側１２への１方向しか実現できなかった。サーバ側１２からクライアント側１１にメッセージを送信しようとしても、ファイアウオール１３によりブロックされるので、クライアント側１１には到達しないという問題があった。

【０００４】また、送受信の記録を相手側に提示するだけでは、確かに送受信をしたという証拠とならないので、送信事実を受信側が否認したり、受信事実を送信側が否認したりすることができてしまうという問題もあった。このような、送信事実の否認、受信事実の否認は、特にインターネットを介して金銭取引を行う場合等において重大である。

【０００５】本発明の目的は、上記従来技術における問題に鑑み、片方向のリクエスト／レスポンス型の通信プ

ロトコルを用いて、後述する再送手順により双方向の送達確認を可能にする通信方法を提供することにある。

【０００６】本発明の他の目的は、上記通信方法において、メッセージに一意の識別子を付加したり、送受信されたメッセージに対して署名を作成し、これを交換することにより、送受信が確かに行なわれたことを証拠として残し、それにより、送信事実、受信事実の否認を防止することにある。

【０００７】本発明のさらに他の目的は、ＸＭＬ(Extensive Markup Language)を用いてメッセージを構築することにより、メッセージの一意性を保証する処理の自動化を容易にすることにある。

【０００８】

【課題を解決するための手段】上記の目的を達成するために、本発明の第１の態様により、レスポンス側からリクエスト側に向けてメッセージを送信する際は、リクエスト側がメッセージの受信要求を行ない、リクエスト側がメッセージを受信したことを、新たな通信でレスポンス側に受信通知するまで、レスポンス側は同じメッセージをリクエスト側に送信し続けることにより、双方向でメッセージの送受信を可能にした双方向通信手法が提供される。

【０００９】リクエスト側からの受信要求に応じてであれば、レスポンス側はメッセージをリクエスト側に送信できるので、リクエスト側からとレスポンス側からの双方向からメッセージの送信が可能になる。

【００１０】本発明の第２の態様により、上記第１の態様において、送信するメッセージ内に一意の識別子を付加し、受信側で重複チェックを行なうことにより、同じメッセージを重複して受信することを防ぐようにした。

【００１１】本発明の第３の態様により、上記第２の態様において、メッセージの形式にＸＭＬを利用し、メッセージとは異なる特定のネームスペースでメッセージ内に識別子を付加することで、メッセージの形式を変更せずに、同じメッセージを重複して受信することを防ぐようにした。

【００１２】本発明の第４の態様により、上記第１の態様において、メッセージの一意性は任意の形で実現し、メッセージの一意性を検証する際に、メッセージのダイジェスト値を利用することで、同じメッセージを重複して受信することを防ぐようにした。

【００１３】上記第２から第４の態様により、同じメッセージを重複して受信することを防ぐことができるので、通信効率を向上させることができる。

【００１４】本発明の第５の態様により、上記第２から第４の態様のいずれかにおいて、メッセージの送信側が送信するメッセージに対して送信メッセージ用電子署名を付加し、この送信メッセージ用署名を受信側が保管することによって送信側による送信事実の否認を防止することができるようにすることと、メッセージの受信側が

メッセージの受信に応答して送信するメッセージ受信通知に、受信通知用電子署名を付加し、この受信通知用電子署名を、送信側が保管することによって受信側による受信事実の否認を防止することができるようにすることと、の少なくとも一方を採用する。

【００１５】これにより、送信側による送信事実の否認や受信側による受信事実の否認を防止することができる。

【００１６】

【発明の実施の形態】以下、本発明の実施の形態を図面を参照しながら説明する。以下の説明で同一参照番号は同一のものを表し、図１に示したファイアウオール１３は図面の簡単化のために図示を省略してある。

【００１７】図２は、本発明の第１の実施の形態によりリクエスト側１１からレスポンス側１２にメッセージの送信する場合のデータの流れを示すブロック図である。リクエスト側１１の例としては、商品の売主や買主等、サーバを持たないクライアントがある。レスポンス側１２の例としては、申し込み書等をデータベースに保管しているセンタの役割を果たすサーバがある。

【００１８】図３は図２に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【００１９】図２および図３で説明するメッセージの送信自体は、従来の片方向通信で行われていたものである。

【００２０】図２及び図３において、リクエスト側１１では、ステップＳ３１で１つ又は複数のメッセージを作成し、ステップＳ３２でそのメッセージを保管し、ステップＳ３３で送信したいメッセージ２１をレスポンス側１２に送信する。メッセージ２１には受信通知の返信を要求するリクエストが含まれている。

【００２１】レスポンス側１２では、ステップＳ３４でそのメッセージを受信し、ステップＳ３５で受信したメッセージ中のＩＤ（識別子）と同じＩＤがレスポンス側１２に存在するかを検索する。その検索の結果ステップＳ３６で同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ３７にて受信メッセージを保管し、ステップＳ３８で受信通知を作成する。ステップＳ３６の検索の結果、同じＩＤがレスポンス側１２に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ３８にて受信通知を作成する。次いでステップＳ３９にて受信通知２２をリクエスト側１１に向けて返信する。受信通知２２には、リクエストに対するレスポンスである旨の情報が含まれている。

【００２２】リクエスト側１１ではステップＳ４０にて所定時間間隔毎に上記受信通知２２の受信を監視し、ステップＳ４１にて対応するメッセージを送信したか、即ちステップＳ３３にてリクエスト側１１から送信された

5

メッセージがレスポンス側１２に正常に受信されたかを受信通知２２の有無により判断し、未だ受信通知２２を受信していない場合は対応するメッセージ２１を送信していないと判断してステップＳ３３に戻りメッセージ２１をレスポンス側１２に再送する。

【００２３】ステップＳ４１の判断でメッセージ２１が送信済みと判定された場合は、ステップＳ４２にてリクエスト側１１に保管されている対応するメッセージ２１を削除する。

【００２４】上記図２及び図３により説明した、リクエスト側１１からレスポンス側１２側への片方向メッセージの送信自体は従来から可能であった。

【００２５】図４は本発明の第１の実施の形態により双方向通信を可能にする場合のデータの流れを示すブロック図であり、図５は図４に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【００２６】図４及び図５において、レスポンス側１２では、ステップＳ５１で１つ又は複数のメッセージを作成し、ステップＳ５２でそのメッセージを保管しておく。

【００２７】リクエスト側１１では、ステップＳ５３にて受信要求４１をレスポンス側１２に送信する。レスポンス側１２はステップＳ５４でこの受信要求４１を受信し、ステップＳ５５でその受信要求に応ずるメッセージ４２を保管されているメッセージの中から検索する。受信要求に応ずるメッセージ４２が存在すればステップＳ５６にてそのメッセージ４２をリクエスト側１１に送信する。

【００２８】リクエスト側１１ではそのメッセージ４２をステップＳ５７で受信し、ステップＳ５８で受信したメッセージ中のＩＤ（識別子）と同じＩＤがリクエスト側１１に存在するかを検索する。検索した結果、ステップＳ５９で同じＩＤがリクエスト側１１に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ６０にて受信メッセージを保管し、ステップＳ６１で受信通知４３を作成する。ステップＳ５９の検索の結果、同じＩＤがリクエスト側１１に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ６１にて受信通知４３を作成する。次いでステップＳ６２にて受信通知４３をレスポンス側１２に向けて返信する。

【００２９】レスポンス側１２ではステップＳ６３にて所定時間間隔毎に上記受信通知４３の受信を監視し、ステップＳ６４にて対応するメッセージを送信したか、即ちステップＳ６３にてレスポンス側１２から返信されたメッセージがリクエスト側１１に正常に受信されたかを受信通知４３の有無により判断し、受信通知４３が受信されていればメッセージ４２がリクエスト側１１からレスポンス側１２に送信済みと判定され、ステップＳ６５にてレスポンス側１２に保管されている、受信要求４１に対応するメッセージ４２を削除する。

6

【００３０】未だ受信通知４３を受信していない場合はステップＳ６６にて処理を終了する。

【００３１】リクエスト側１１からレスポンス側１２には受信要求４１が定期的に送信されるので、このメッセージ４２の送信はリクエスト側１１から受信通知４３が到着するまで、定期的に繰り返し行われる。受信通知４３がリクエスト側１１から送られなければ、レスポンス側１２ではメッセージ４２が確かにリクエスト側１１に届いたかどうか不明であるが、本発明の実施の形態により、メッセージ４２の受信に応答して受信通知４３をリクエスト側１１からレスポンス側１２に返信するようにしたことにより、レスポンス側１２では確実にメッセージ４２がリクエスト側に届いたことを認識でき、結果的に双方向のメッセージの遣り取りが可能になる。

【００３２】このように、リクエスト側１１に設けられたフィアウオールでリクエスト側１１からのみ外部にメッセージを送ることができるシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する必要性及びレスポンス側１２から送られたメッセージがリクエスト側１１に確実に受信されたことを認識できる必要性は、特に金融に関する通信や企業内のクライアントと企業外の別の企業との間の文書の通信や、レスポンス側１２で発行する何らかの申し込み書等で必要になる。例えば、申し込み書の配信等のように、レスポンス側１２から送られるウェブブラウザの画面中にメッセージを含ませたいという要求がある。従来ではこの場合、レスポンス側１２からリクエスト側１１へのメッセージの配信は不可能であったが、本発明の実施の形態によれば上記のように確実にメッセージをレスポンス側１２からリクエスト側１１に配信することが可能になる。

【００３３】尚、メッセージの送受信にはＳＯＡＰ（Simple Object Access Protocol）等のプロトコルを用いて実現できる。

【００３４】図６は本発明の第２の実施の形態によりメッセージ中の定まったフォーマットの中の特定の場所にメッセージの識別子（ＩＤ）を埋め込む方法を説明する図である。図において、ルートタグ<Order>からルートタグ</Order>までが定まったフォーマットの注文書のメッセージであり、このメッセージ中のmessageIdという２つのタグの間にＩＤとして"12345678"が埋め込まれている。受信側ではこのメッセージのフォーマットを知っているので受信メッセージ中のＩＤを自分が持つＩＤと照合することにより受信済みメッセージかどうかを判断する。図示例はＸＭＬで記載された注文書のメッセージを例としてあげてあるが、定まったフォーマットのメッセージであればどのような言語で記載されていてもよい。これにより、受信側ではプログラムが解釈可能なフォーマットからメッセージＩＤを検索可能になる。

【００３５】図７は本発明の第３の実施の形態によりメ

7

ッセージ内にメッセージ内容とは別にネームスペースでメッセージＩＤを埋め込む方法を説明する図である。図において、ルートタグ<Order>からルートタグ</Order>までは図６と同じく注文書のメッセージであるが、このメッセージ中のプレフィックスssrs：とその後の属性名messageIdの後に="12345678"というメッセージＩＤを埋め込んである。これにより、受信側ではプログラムによりプレフィックスの属性を探せば、ＩＤを見つけることができ、メッセージの一意性を検証することができる。

【００３６】図７の例では、ルートタグとして<Order>を用いているが、プレフィクスとその属性ssrs:messageId=の後にメッセージＩＤを埋め込めばよいので、ルートタグが何であってもＩＤを見つけることができる。また、メッセージのフォーマットが何であってもよい。

【００３７】図８は本発明の第４の実施の形態により、メッセージのダイジェストを利用してメッセージの一意性を検証する方法を説明する図である。この場合は、ルートタグ<Order>からルートタグ</Order>までのメッセージの中の任意の場所に<orderId>order012345678</orderId>というＩＤを含む記号を挿入した後に、ＳＨＡ１，ＭＤ５等のアルゴリズムに従ってこのメッセージをダイジェストに圧縮して送信する。受信側でこのダイジェストを所定のアルゴリズムを用いて計算する。受信メッセージの内容及びＩＤのいずれかが、既に受信したメッセージの内容及びＩＤのいずれかと異なれば、ダイジェストの計算値も異なる、受信メッセージの内容及びＩＤのいずれもが、既に受信したメッセージの内容及びＩＤのいずれもと一致していれば、ダイジェストの計算値も一致する。これによっても、受信メッセージの一意性の検証が可能になる。図８においても、メッセージは注文書に限定されずどのような内容でもよい。また、メッセージのフォーマットも任意でよい。

【００３８】図９は本発明の第５の実施の形態により送信者による送信の事実を後に送信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図１０は図９に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【００３９】図１０と図３との相違点は、図１０においてはリクエスト側１１でステップＳ１０２にて送信署名を作成し、レスポンス側１２でステップＳ１０６及びステップＳ１０７で送信署名の検証をし、ステップＳ１１１でメッセージの外に署名も保管し、リクエスト側１１でステップＳ１１５にて送信署名エラーが返信されたかを判断することが追加されていることである。

【００４０】より詳細には、図９及び図１０において、リクエスト側１１では、ステップＳ１０１で１つ又は複数のメッセージを作成し、ステップＳ１０２で送信者だ

8

けが作成できる送信者署名を作成し、ステップＳ１０３でその署名をメッセージとともに保管し、ステップＳ１０４で送信したいメッセージ２１ａをレスポンス側１２に送信する。メッセージ２１には受信通知の返信を要求するリクエストと署名とが含まれている。

【００４１】レスポンス側１２では、ステップＳ１０５でそのメッセージを受信し、ステップＳ１０６で送信署名を送信者の公開鍵で検証する。ステップＳ１０７でその検証結果が不正な署名であれば、ステップＳ１０８に進んで署名検証エラーを受信通知２２に載せてリクエスト側１１に返信する。ステップＳ１０７で署名の検証結果が正しければ、ステップＳ１０９にて図６から図８のいずれかの方法により、受信したメッセージ中のＩＤ（識別子）と同じＩＤがレスポンス側１２に存在するかを検索する。その検索の結果ステップＳ１１０にて同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ１１１にて受信メッセージと共に送信者の署名９０をデータベース９１に保管し、ステップＳ１１２で受信通知を作成する。ステップＳ１１０の検索の結果、受信メッセージがレスポンス側１２に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ１１２にて受信通知を作成する。次いでステップＳ１１３にて受信通知２２をリクエスト側１１に向けて返信する。受信通知２２には、リクエストに対するレスポンスである旨の情報が含まれている。

【００４２】リクエスト側１１ではステップＳ１１４にて所定時間間隔毎に上記受信通知２２の受信を監視し、ステップＳ１１５にて送信署名エラーが返信されたかを判定し、送信署名エラーが返信された場合はステップＳ１１６にてエラー処理をする。

【００４３】ステップＳ１１５の判断で送信署名エラーでないと判定されると、ステップＳ１１７にて対応するメッセージを送信したかに正常に受信されたか、即ちステップＳ１０４にてリクエスト側１１から送信されたメッセージがレスポンス側１２を受信通知２２の有無により判断し、未だ受信通知２２を受信していない場合は対応するメッセージ２１ａを送信していないと判断してステップＳ１０４に戻りメッセージ２１ａをレスポンス側１２に再送する。

【００４４】ステップＳ１１７の判断でメッセージ２１ａが送信済みと判定された場合は、ステップＳ１１８にてリクエスト側１１に保管されている対応するメッセージ２１を削除する。

【００４５】これにより、リクエスト側１１が送信した事実を否認しても、レスポンス側１２のデータベース９１にはリクエスト側１１でのみ作成可能な署名が格納されているので、リクエスト側１１による送信事実の否認をレスポンス側１２は拒否できる。

【００４６】図１１は、図９に示したシステムにおい

て、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【００４７】図１１と図５との相違点は、図１１においては、レスポンス側１２でステップＳ１２２にて送信署名を作成し、リクエスト側１１ではステップＳ１２９、ステップＳ１３０及びステップＳ１３１で送信署名検証を行い、ステップＳ１３４でメッセージの外に署名も保管し、レスポンス側１２ではステップＳ１３８及びステップＳ１３９で送信署名エラーかを判断し処理することが追加されていることである。

【００４８】より詳細には、図９及び図１１において、レスポンス側１２では、ステップＳ１２１で１つ又は複数のメッセージを作成し、ステップＳ１２２で送信者のみが作成できる送信書名を作成する。次いでステップＳ１２３でその送信署名をメッセージとともに保管しておく。

【００４９】リクエスト側１１では、ステップＳ１２４にて受信要求４１をレスポンス側１２に送信する。レスポンス側１２はステップＳ１２５でこの受信要求４１を受信し、ステップＳ１２６でその受信要求に応ずるメッセージ４２を保管されているメッセージの中から検索する。受信要求に応ずるメッセージ４２ａが存在すればステップＳ１２７にてそのメッセージ４２ａに署名９２を含ませてリクエスト側１１に送信する。

【００５０】リクエスト側１１ではそのメッセージ４２ａをステップＳ１２８で受信し、ステップＳ１２９で受信したメッセージ中の送信署名が正しいかどうかを送信者の公開鍵で検証する。正しくなければステップＳ１３１で署名検証エラーをレスポンス側１２に返信する。正しければステップＳ１３２及びステップＳ１３３にて受信メッセージ中のＩＤ（識別子）と同じＩＤがリクエスト側１１に存在するかを検索する。検索した結果、同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ１３４にて受信メッセージを署名９２とともにデータベース９３に保管し、ステップＳ１３５で受信通知４３を作成する。ステップＳ１３３の検索の結果、同じＩＤがリクエスト側１１に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ１３５にて受信通知４３を作成する。次いでステップＳ１３６にてその受信通知をレスポンス側１２に向けて返信する。

【００５１】レスポンス側１２ではステップＳ１３７にて所定時間間隔毎に上記受信通知４３の受信を監視し、ステップＳ１３８にて送信署名がエラーかを判断する。エラーであれば、ステップＳ１３９にてエラー処理をする。エラーでなければステップＳ１４０にて対応するメッセージを送信したか、即ちステップＳ１２７にてレスポンス側１２から返信されたメッセージがリクエスト側１１に正常に受信されたかを受信通知４３の有無により判断し、メッセージ４２ａが送信済みと判定された場合は、ステップＳ１４１にてレスポンス側１２に保管されている対応するメッセージ４２ａを削除する。

【００５２】未だ受信通知４３を受信していない場合はステップＳ１４２にて処理を終了する。

【００５３】これにより、レスポンス側１２が送信した事実を否認しても、リクエスト側１１のデータベース９２にはレスポンス側１２でのみ作成可能な署名９２が格納されているので、レスポンス側１２による送信事実の否認をリクエスト側１１は拒否できる。

【００５４】図１２は本発明の第６の実施の形態により受信者による受信の事実を後に受信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図１３は図１２に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【００５５】図１３と図３との相違点は、図１３においてはレスポンス側１２でステップＳ１５８にて受信署名を作成し、リクエスト側１１でステップＳ１６３及びステップＳ１６４で受信署名の検証をし、ステップＳ１６５で受信署名を保管することが追加されていることである。

【００５６】より詳細には、図１２及び図１３において、リクエスト側１１では、ステップＳ１５１で１つ又は複数のメッセージを作成し、ステップＳ１５２でメッセージを保管し、ステップＳ１５３で送信したいメッセージ２１をレスポンス側１２に送信する。メッセージ２１には受信通知の返信を要求するリクエストが含まれている。

【００５７】レスポンス側１２では、ステップＳ１５４でそのメッセージを受信し、ステップＳ１５５で図６から図８のいずれかの方法により、受信したメッセージ中のＩＤ（識別子）と同じＩＤがレスポンス側１２に存在するかを検証する。その検証の結果ステップＳ１５６にて同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ１５７にて受信メッセージを保管し、ステップＳ１５８で受信者だけが作成できる受信署名１２０を作成する。ステップＳ１５６の判断の結果、同じＩＤがレスポンス側１２に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ１５９にて受信通知を作成する。次いでステップＳ１６０にて受信通知２２ａをリクエスト側１１に向けて返信する。受信通知２２ａには、リクエストに対するレスポンスである旨の情報に加えて受信署名１２０も含まれている。

【００５８】リクエスト側１１ではステップＳ１６１に

て所定時間間隔毎に上記受信通知２２の受信を監視し、ステップＳ１６２にて対応するメッセージを送信したか、即ち、ステップＳ１５３にてリクエスト側１１から送信したメッセージがレスポンス側に正常に受信されたかを受信通知２２ａの有無により判断し、未だ受信通知２２ａを受信していない場合は対応するメッセージ２１を送信していないと判断してステップＳ１５３に戻りメッセージ２１をレスポンス側１２に再送する。

【００５９】ステップＳ１６４の判断でメッセージ２１が送信済みと判定された場合は、ステップＳ１６５にて受信署名をデータベース１２１に補完し、ステップＳ１６６にてリクエスト側１１に保管されている対応するメッセージ２１を削除する。

【００６０】これにより、レスポンス側１２が受信した事実を否認しても、リクエスト側１１のデータベース１２１にはレスポンス側１２でのみ作成可能な署名が格納されているので、レスポンス側１２による受信事実の否認をリクエスト側１１は拒否できる。

【００６１】図１４は、図１２に示したシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【００６２】図１４と図５との相違点は、図１４においては、リクエスト側１１でステップＳ１８１にて受信署名を作成し、レスポンス側１２ではステップＳ１８６及びステップＳ１８７で送信署名検証を行い、ステップＳ１８８で受信署名を保管することである。

【００６３】より詳細には、図１２及び図１４において、レスポンス側１２では、ステップＳ１７１で１つ又は複数のメッセージを作成し、ステップＳ１７２でそのメッセージを保管しておく。

【００６４】リクエスト側１１では、ステップＳ１７３にて受信要求４１をレスポンス側１２に送信する。レスポンス側１２はステップＳ１７４でこの受信要求４１を受信し、ステップＳ１７５でその受信要求に応ずるメッセージ４２を保管されているメッセージの中から検索する。受信要求に応ずるメッセージ４２が存在すればステップＳ１７６にてそのメッセージ４２をリクエスト側１１に送信する。

【００６５】リクエスト側１１ではそのメッセージ４２をステップＳ１７７で受信し、ステップＳ１７８で受信したメッセージ中のＩＤ（識別子）と同じＩＤがリクエスト側１１に存在するかを検証する。検証した結果、ステップＳ１７９で同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ１８０にて受信メッセージを保管し、ステップＳ１８１で受信者のみが作成できる受信書名１２２を作成し、ステップＳ１８４で受信署名１２２を含む受信通知４３ａを作成する。ステップＳ１７９の

検索の結果、同じＩＤがリクエスト側１１に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ１８１で受信者のみが作成できる受信書名１２２を作成し、ステップＳ１８４で受信署名１２２を含む受信通知４３ａを作成する。次いでステップＳ１８３にてその受信通知４３ａをレスポンス側１２に向けて返信する。

【００６６】レスポンス側１２ではステップＳ１８４にて所定時間間隔毎に上記受信通知４３ａの受信を監視し、ステップＳ１８５にて対応するメッセージを送信したか、即ち、ステップＳ１７６にてレスポンス側１２から送信されたメッセージがリクエスト側１１に正常に受信されたかを受信通知４３ａの有無により判断し、メッセージ４２が送信済みと判定された場合は、ステップＳ１８６にて受信署名を受信者の公開鍵で検証する。ステップＳ１８７の判断で正しい受信署名であると判定されると、ステップＳ１８８にて受信署名をデータベース１２３に保管し、ステップＳ１８９にてレスポンス側１２に保管されている対応するメッセージ４２を削除する。

【００６７】ステップＳ１８５の判断で未だ受信通知４３を受信していない場合、又はステップＳ１８７の判断で正しい受信署名ではないと判定された場合はステップＳ１９０にて処理を終了する。

【００６８】これにより、リクエスト側１１が受信した事実を否認しても、レスポンス側１２のデータベース１２３にはリクエスト側１１でのみ作成可能な受信署名１３２が格納されているので、リクエスト側１１による受信事実の否認をレスポンス側１２は拒否できる。

【００６９】図１５は本発明の第７の実施の形態により送受信者による送受信の事実を後に送受信者が否認することを防止する方法におけるデータの流れを説明するブロック図であり、図１６は図１５に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合の処理の流れを説明するフローチャートである。

【００７０】図１６は図１０と図１３とを融合したものである。

【００７１】より詳細には、図１５及び図１６において、リクエスト側１１では、ステップＳ１９１で１つ又は複数のメッセージを作成し、ステップＳ１９２で送信者だけが作成できる送信署名を作成し、ステップＳ１９３でその送信署名をメッセージとともにを保管し、ステップＳ１９４で送信したいメッセージ２１ａをレスポンス側１２に送信する。メッセージ２１ａには受信通知の返信を要求するリクエストと送信署名とが含まれている。

【００７２】レスポンス側１２では、ステップＳ１９５でそのメッセージを受信し、ステップＳ１９６で送信署名を送信者の公開鍵で検証する。ステップＳ１９７でその検証結果が不正な署名であれば、ステップＳ１９８に

13

進んで署名検証エラーを受信通知２２ａに載せてリクエスト側１１に返信する。ステップＳ１９７で著名の検証結果が正しければ、ステップＳ１９９にて図６から図８のいずれかの方法により、受信したメッセージ中のＩＤ（識別子）と同じＩＤがレスポンス側１２に存在するかを検索する。その検索の結果ステップＳ２００にて同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ２０１にて受信メッセージと共に送信者の署名９０をデータベース９１に保管し、ステップＳ２０２で受信者のみが作成できる受信書名１２０を作成する。次いでステップＳ２０４にて受信通知２２ａをリクエスト側１１に向けて返信する。受信通知２２ａには、リクエストに対するレスポンスである旨の情報と受信署名とが含まれている。

【００７３】リクエスト側１１ではステップＳ２０５にて所定時間間隔毎に上記受信通知２２ａの受信を監視し、ステップＳ２０６にて送信署名エラーが返信されたかを判定し、送信署名エラーが返信された場合はステップＳ２０７にてエラー処理をする。

【００７４】ステップＳ２０６の判断で送信署名エラーでないと判定されると、ステップＳ２０８にて対応するメッセージを送信したか、即ち、ステップＳ１９４にてリクエスト側１１から送信したメッセージがレスポンス側１２に正常に受信されたかを受信通知２２ａの有無により判断し、未だ受信通知２２ａを受信していない場合は対応するメッセージ２１ａを送信していないと判断してステップＳ１９４に戻りメッセージ２１ａをレスポンス側１２に再送する。

【００７５】ステップＳ２０６の判断でメッセージ２１ａが送信済みと判定された場合は、ステップＳ２０９にて受信通知２２ａ内の受信署名を受信者の公開鍵で検証する。この検証の結果ステップＳ２１０で正しくない受信署名であると判定されると、リクエスト側１１からレスポンス側１２にメッセージ２１ａが未だ送信されていないものと判断して、ステップＳ１９４に戻りメッセージ２１ａをレスポンス側１２に再送する。

【００７６】ステップＳ２１０の検証で正しい受信署名であると判定されると、ステップＳ２１１にて受信署名をデータベース１２１に保管し、次いでステップＳ２１２にてリクエスト側１１に保管されている対応するメッセージ２１を削除する。

【００７７】これにより、リクエスト側１１が送信した事実を否認しても、レスポンス側１２のデータベース９１にはリクエスト側１１でのみ作成可能な送信署名が格納されているので、リクエスト側１１による送信事実の否認をレスポンス側１２は拒否できるとともに、レスポンス側１２が受信した事実を否認しても、リクエスト側１１のデータベース１２１にはレスポンス側１２でのみ作成可能な受信署名が格納されているので、レスポンス側１２による受信事実の否認をリクエスト側１１は拒否できる。

14

【００７８】図１７は、図１５に示したシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、送受信事実の送受信者による否認を防止する処理の流れを説明するフローチャートである。

【００７９】図１７は図１１と図１１と図１４を融合したものである。

【００８０】より詳細には、図１５及び図１７において、レスポンス側１２では、ステップＳ２２０で１つ又は複数のメッセージを作成し、ステップＳ２２１で送信者のみが作成できる送信書名を作成する。次いでステップＳ２２２でその送信署名をメッセージとともに保管しておく。

【００８１】リクエスト側１１では、ステップＳ２２３にて受信要求４１をレスポンス側１２に送信する。レスポンス側１２はステップＳ２２４でこの受信要求４１を受信し、ステップＳ２２５でその要求に応ずるメッセージ４２ａを保管されているメッセージの中から検索する。受信要求に応ずるメッセージ４２ａが存在すればステップＳ２２６にてそのメッセージ４２ａに署名９２を含ませてリクエスト側１１に送信する。

【００８２】リクエスト側１１ではそのメッセージ４２ａをステップＳ２２７で受信し、ステップＳ２２８で受信したメッセージ中の送信署名が正しいかどうかを送信者の公開鍵で検証する。正しくなければステップＳ２３０で署名検証エラーをレスポンス側１２に返信する。正しければステップＳ２３１及びステップＳ２３２にてメッセージ４２ａ中のＩＤ（識別子）と同じＩＤがリクエスト側１１に存在するかを検索する。検索した結果、同じＩＤがレスポンス側１２に存在しなければ、今回受信したメッセージは新たなメッセージなので、ステップＳ２３３にて受信メッセージを署名９２とともにデータベース９３に保管し、ステップＳ２３４で受信者のみが作成できる受信署名１２２を作成し、ステップＳ２３５で受信署名１２２を含む受信通知４３ａを作成する。次いでステップＳ２３６で受信通知４３ａをレスポンス側１２に送信する。

【００８３】ステップＳ２３２の検索の結果、同じＩＤがリクエスト側１１に存在すれば、受信済みのメッセージなので、今回受信したメッセージは保管せずに、ステップＳ２３５にて受信署名を作成する。

【００８４】レスポンス側１２ではステップＳ２３７にて所定時間間隔毎に上記受信通知４３ａの受信を監視し、ステップＳ２３８にて送信署名がエラーかを判断する。エラーであれば、ステップＳ２３９にてエラー処理をする。エラーでなければステップＳ２４０にて対応するメッセージを送信したか、即ち、ステップＳ２２６にてレスポンス側１２から送信したメッセージがリクエス

15

ト側１１に正常に受信されたかを受信通知４３ａの有無により判断し、メッセージ４２ａが送信済みと判定された場合は、ステップＳ２４１にて受信者の公開鍵により受信署名を検証し、ステップＳ２４２で正しい署名であればステップＳ２４３で受信署名１２２をデータベース１５１に保管し、ステップＳ２４４でレスポンス側１２に保管されている対応するメッセージ４２ａを削除する。

【００８５】ステップＳ２４０にて未だ対応するメッセージを送信していないと判定された場合又はステップＳ２４２にて正しい受信署名ではないと判定された場合は、ステップＳ２４５にて処理を終了する。

【００８６】これにより、レスポンス側１２が送信した事実を否認しても、リクエスト側１１のデータベース９３にはレスポンス側１２でのみ作成可能な送信署名が格納されているので、レスポンス側１２による送信事実の否認をリクエスト側１１は拒否できるとともに、リクエスト側１１が受信した事実を否認しても、レスポンス側１２のデータベース１５１にはリクエスト側１１でのみ作成可能な受信署名が格納されているので、リクエスト側１１による受信事実の否認をレスポンス側１２は拒否できる。以上に記載した各実施の形態において、受信したメッセージ及び署名の少なくとも一方の保管が正常に完了しない場合がある。その原因としては、メッセージや署名を保管するバッファの領域が不足している場合や、そのバッファに書き込み禁止フラグが立てられている等によりそのバッファへの書き込み権がない場合や、保存先を間違えた場合や、保存先がデータベースの場合でそのデータベースが起動していない場合等がある。バッファの領域が不足しているためにメッセージや署名の保管ができなかった場合は、以下に記載する本発明の第８及び第９の実施の形態により保管可能になる場合がある。図１８は、図３におけるステップＳ３７又は図５におけるステップＳ６０におけるメッセージの保管が成功しなかった場合の本発明の第８の実施の形態によるメッセージ保管方法を説明するフローチャートである。同図において、ステップＳ２５０にて署名がない受信メッセージの保管動作を行う。このステップは図３におけるステップＳ３７又は図５におけるステップＳ６０の動作である。次いでステップＳ２５１にてメッセージ保存エラーかどうかを判定し、エラーであればステップＳ２５２に進んでエラーの原因がメッセージ保存用バッファの領域不足かどうかを判定する。領域不足であればステップＳ２５３にてそのメッセージ保存領域の中の最も古いメッセージから順に少なくとも一つのメッセージを削除し、ステップＳ２５４にて再度領域不足かを判定する。この判定で依然として領域不足であれば、メッセージ保存に失敗したことを示すメッセージ保存エラーをメッセージの送信側に返信する。ステップＳ２５４にて領域不足でなければステップＳ２５０に戻りメッセージを保管

16

する。ステップＳ２５１にてメッセージ保存エラーでなければステップＳ２５５にて受信通知を作成する。ステップＳ２５５は図３におけるステップＳ３８又は図５におけるステップＳ６１に相当する。図１８に示したメッセージ保管エラーの救済方法は、図１３におけるステップＳ１５７又は図１４におけるステップＳ１８０のメッセージ保管が出来なかった場合にも同様に適用できる。また、メッセージに替えて受信した署名の保管が出来なかった場合にも同様に適用できる。即ち、図１３のステップＳ１６５、図１４のステップＳ１８８、図１６のステップＳ２１１、図１７のステップＳ２４３において受信署名の保管に失敗した場合に、図１８に示したフローチャートにおける「メッセージ」を「署名」に変更したフローチャート（図示省略）により、署名保管のエラーを救済できる場合がある。図１９は、図１６におけるステップＳ２０１又は図１７におけるステップＳ２３３におけるメッセージ及び署名の保管が成功しなかった場合の本発明の第９の実施の形態によるメッセージ及び署名の保管方法を説明するフローチャートである。同図において、ステップＳ２６０にて受信メッセージ及び署名の保管動作を行う。このステップは図１６におけるステップＳ２０１又は図１７におけるステップＳ２３３の動作である。次いでステップＳ２６１にてメッセージ及び署名の少なくとも一方の保存エラーかどうかを判定し、エラーであればステップＳ２６２に進んでエラーの原因がメッセージ及び署名の保存領域の領域不足かどうかを判定する。領域不足であればステップＳ２６３にてそのメッセージ及び署名の保存領域の中の最も古いメッセージから順に少なくとも一つのメッセージを削除し、ステップＳ２６４にて再度領域不足かを判定する。この判定で依然として領域不足であれば、ステップＳ２６５にてそのメッセージ及び署名の保存領域の中の最も古い署名を削除し、ステップＳ２６６にて再度領域不足かを判定する。この判定で依然として領域不足であれば、署名検証エラーであることを示すメッセージを送信側に返信する。ステップＳ２６６にて領域不足でなければステップＳ２６０に戻りメッセージ及び署名を保管する。ステップＳ２６１にてメッセージ又は署名の保存エラーでなければステップＳ２６８受信署名を作成し、ステップＳ２６９にて受信通知を作成する。ステップＳ２６８及びステップＳ２６９は図１６におけるステップＳ２０２及びステップＳ２０３又は図１７におけるステップＳ２３４及びステップＳ２３５に相当する。図１９に示した署名付きメッセージ保管エラーの救済方法は、図１０におけるステップＳ１１１、図１１におけるステップＳ１３４、図１６におけるステップＳ２０１にも同様に適用できる。上記のように、バッファの容量不足により受信したメッセージや署名の保管に失敗した場合にも、古いメッセージや署名をバッファから削除することにより確実に受信メッセージや署名を保存できる。以上に記載した

17

各実施の形態において用いられるメッセージの形式の一例を以下に記載する。以下の例ではメッセージはアプリケーション間通信の取り決めを定義したExtensible Markup Language(XML)形式によりSOAP の標準に従って作成されている。図２０はリクエスト側からレスポンス側に対してメッセージの送信要求をする場合のメッセージの内容の一例である。このメッセージではヘッダ内にメッセージタイプとして送信要求を示すRequestを記述する。このメッセージは図５のステップＳ５３、図１１のステップＳ１２４、図１４のステップＳ１７３又は図１７のステップＳ２２３で使用される。図２０において、<SOAP:Envelope と</SOAP:ENVELOPE>との間の情報の中で、最初の行にxmls:SOAP=…と記載されているのは、このメッセージが SOAP を標準とする XML 形式で記載されたものであることを示している。<SOAP:Header> と </SOAP:Header> との間の <rm:Message Header…と</rm:Message Header>の間がこのメッセージのヘッダである。メッセージヘッダには、メッセージの送信元と、送信先と、メッセージのサービスの種類と、メッセージの種類とが記載されている。この例では送信元は<rm:From >で始まる行に記載されているように requester@anyuri.com となっておりリクエスト側である。また、送信先は<rm:To>で始まる行に記載されているようにresponder@someeuri.comとなっておりレスポンス側である。また、サービスの種類は<rm:Service>で始まる行にItem Quote Service と記載されているように、問い合わせのサービスである。さらにメッセージの種類は<rm: Message Type>で始まる行に Request と記載されているように要求メッセージである。このメッセージの送信要求にはメッセージ本体の情報がないので、<SOAP:Body>と</SOAP:Body>との間には何も記載されていない。図２１は署名無し送信メッセージの一例を示す図である。このメッセージは図３のステップＳ３３におけるリクエスト側からのメッセージ送信に使用される。同図において、<SOAP:Envelope と</SOAP:ENVELOPE>との間の情報の中で、最初の行は図２０と同じである。また、<rm:Message Header…>と</rm:Message Header>との間のヘッダにおける、送信元、送信先、サービスの種類も図２０の例と同じである。この他にメッセージ種類、メッセージＩＤ、送信日時等が記載されている。メッセージの種類は<rm: Message Type>で始まる行にMessage と記載されているようにメッセージである。また、メッセージＩＤとして<rm: Message Id>の行に20020907-045261-0450@anyuri.com が記載されている。さらに、送信日時として<rm: Timestamp>の行に 2002-09-07T10:19:07 と記載されている。<rm:Reliable Message…>と</rm:Reliable Message>の間のヘッダには、送信メッセージの信頼性を確保するための情報が記載されている。即ち、<rm:AckRequested SOAP:must understand...は確認メッセージの形式が SOAP の標準に準拠すべきことを記述したものである。また、<rm:Duplicate Elimination/>は重複受信をしたメッセージを削除するものである。<SOAP:BODY>と</SOAP:BODY>との間にはアプリケーションに依存する情報本体の内容が記載される。図５におけるステップＳ５６で送信されるメッセージも、送信元と送信先が図２２に示したものと逆になるだけで図２２と同様の形式で作成される。図２２は署名無し受信確認メッセージの一例を示す図である。このメッセージは図３のステップＳ３９で使用される。メッセージの形式は図２１に示したものと同様であるので詳細な説明は省略する。サービスの種類が Item Filing Service となっていてファイルスルサービスであること、及びメッセージの種類が Acknowledgement となっていて確認のメッセージであることに着目される。本例では送信元がレスポンス側であり送信先がリクエスト側であるが、送信元をリクエスト側にし送信先をレスポンス側にすれば図５のステップＳ６２で使用するメッセージとなる。図２３から図２５は署名有りの送信メッセージの一例を示す図である。このメッセージは図１０のステップＳ１０４で使用される。このメッセージにおいて、第１行の<SOAP:Envelope>から第１９行の</rm:Reliable Message>までは図２１に示した署名無し送信メッセージと同じである。本例では、その次の行に<wsse:Security xmls:wsse=" から</wsse:BinarySecurity Token>までのタグが存在する。ここでは、ウェブ・サービス・セキュリティ (Web Service Security) (wsse)によりセキュリティを確保している。次のタグである<ds:Signature xmls…から</ds:SignatureValue>までの間で、署名がXML形式で記載されている。署名の対象は、<ds:Xpath>のタグに記載されているように、メッセージヘッダと、リライアブルメッセージ(Reliable Message)と、情報本体(SOAP Body)である。これにより、メッセージヘッダと、リライアブルメッセージと、情報本体に対する改竄を防止できる。このように、送信メッセージに WS-Security 形式の署名を付けることにより、送信者の否認を防止できる。本例においても、送信元をレスポンス側とし送信先をリクエスト側とすれば、図１１のステップＳ１２６で使用するメッセージとなる。図２６から図２８は署名有りの受信確認メッセージの一例を示す図である。このメッセージは図１０のステップＳ１１３、図１３のステップＳ１６０、又は図１６のステップＳ２０４で使用される。メッセージの形式は図２３から図２５に示したものと同様であるので詳細な説明は省略する。サービスの種類が Item Filing Service となっていてファイルスルサービスであること、及びメッセージの種類が Acknowledgement となっていて確認のメッセージであることに着目される。この例においても、受信確認メッセージに WS-Security 形式の署名を付けることにより、メッセージヘッダと、リライアブルメッセージと、情報本体に対する改竄を防止できる。本例においても、送信元をリクエスト側とし、

18

19

送信先をレスポンス側とすることにより図１１のステップＳ１３６、図１４のステップＳ１８３、又は図１７のステップＳ２３６で使用するメッセージとなる。

【００８７】

【発明の効果】上記の通り、本発明によれば、片方向のリクエスト／レスポンス型の通信プロトコルを用いて、双方向の送達確認を実現することができるという効果が得られる。

【００８８】また、送受信されたメッセージに対して署名を作成し、これを交換することで、送受信が確かに行なわれたことを証拠として残し、送信事実、受信事実の否認を防止することができるという効果が得られる。

【００８９】さらに、ＸＭＬ を用いてメッセージを構築することにより、メッセージの一意性を保証する処理の自動化を容易にすることができるという効果が得られる。さらに、受信メッセージの保存がバッファの容量不足によりエラーとなった場合でも、古いメッセージや古い署名を削除することにより、エラーを回避できるという利点もある。

【図面の簡単な説明】

【図１】リクエスト／レスポンス型の同期通信のみを用いる従来のシステムの構成を示すブロック図である。

【図２】本発明の第１の実施の形態によりリクエスト側からレスポンス側にメッセージの送信する場合のデータの流れを示すブロック図である。

【図３】図２に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図４】本発明の第１の実施の形態により双方向通信を可能にする場合のデータの流れを示すブロック図である。

【図５】図４に示したメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図６】本発明の第２の実施の形態によりメッセージ中の定まったフォーマットの中の特定の場所にメッセージの識別子（ＩＤ）を埋め込む方法を説明する図である。

【図７】本発明の第３の実施の形態によりメッセージ内にメッセージ内容とは別にネームスペースでメッセージＩＤを埋め込む方法を説明する図である。

【図８】本発明の第４の実施の形態により、メッセージのダイジェストを利用してメッセージの一意性を検証する方法を説明する図である。

【図９】本発明の第５の実施の形態により送信者による送信の事実を後に送信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図１０】図９に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【図１１】図９に示したシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、送信事実の送信者による否認を防止する処理の流れを説明するフローチャートである。

【図１２】本発明の第６の実施の形態により受信者による受信の事実を後に受信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図１３】図１２に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【図１４】図１２に示したシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、受信事実の受信者による否認を防止する処理の流れを説明するフローチャートである。

【図１５】本発明の第７の実施の形態により送受信者による送受信の事実を後に送受信者が否認することを防止する方法におけるデータの流れを説明するブロック図である。

【図１６】図１５に示したシステムにおいて、リクエスト側１１からレスポンス側１２にメッセージを送信する場合の処理の流れを説明するフローチャートである。

【図１７】図１５に示したシステムにおいて、レスポンス側１２からリクエスト側１１にメッセージを送信する場合において、送受信事実の送受信者による否認を防止する処理の流れを説明するフローチャートである。

【図１８】図３におけるステップＳ３７又は図５におけるステップＳ６０におけるメッセージの保管が成功しなかった場合の本発明の第８の実施の形態によるメッセージ保管方法を説明するフローチャートである。

【図１９】図１６におけるステップＳ２０１又は図１７におけるステップＳ２３３におけるメッセージ及び署名の保管が成功しなかった場合の本発明の第９の実施の形態によるメッセージ及び署名の保管方法を説明するフローチャートである。

【図２０】リクエスト側からレスポンス側に対してメッセージの送信要求をする場合のメッセージの内容の一例である。

【図２１】署名無し送信メッセージの一例を示す図である。

【図２２】署名無し受信確認メッセージの一例を示す図である。

【図２３】署名有りの送信メッセージの一例の一部を示す図である。

【図２４】署名有りの送信メッセージの一例の他の一部を示す図である。

【図２５】署名有りの送信メッセージの一例の更に他の一部を示す図である。

【図２６】署名有りの受信確認メッセージの一例の一部を示す図である。

【図２７】署名有りの受信確認メッセージの一例の他の一部を示す図である。

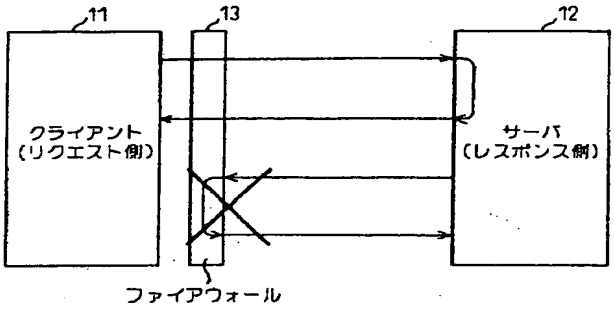
【図２８】署名有りの受信確認メッセージの一例の更に他の一部を示す図である。

【符号の説明】

１１…リクエスト側
１２…レスポンス側
１３…ファイアウォール
２１…メッセージ
２１ａ…メッセージ
２２…受信通知
２２ａ…受信通知
４１…受信要求
４２…メッセージ
４２ａ…メッセージ
４３…受信通知
４３ａ…受信通知
９０…署名
９１…データベース
９２…署名
９３…データベース
１２０…署名
１２１…データベース
１２２…署名
１５１…データベース

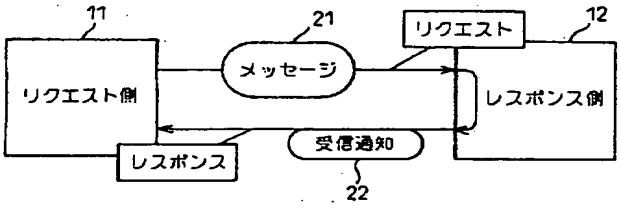
【図１】

図１
片方向のリクエスト／レスポンス型の同期通信



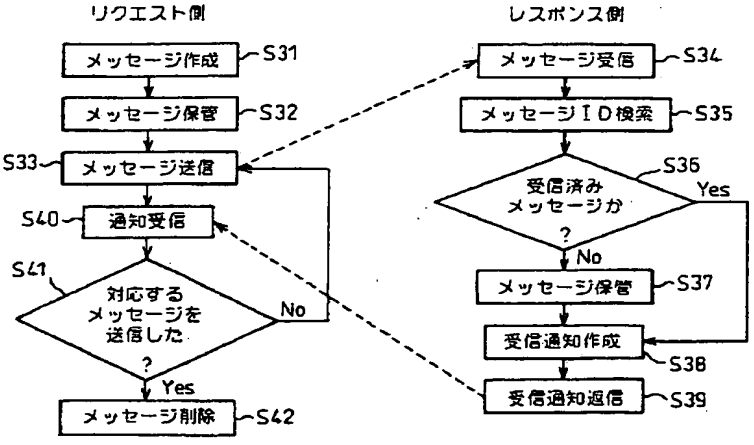
【図２】

図２
リクエスト側からレスポンス側にメッセージを送信する場合



【図３】

図３
リクエスト側からレスポンス側にメッセージを送信する場合の処理の流れ



【図６】

図６　メッセージＩＤを埋め込んだメッセージの例

```
<Order xmlns="http://jp.fujitsu.com/order">
 <messageId>12345678</messageId>
 <userInfo>
  :
 </userInfo>
</Order>
```

【図７】

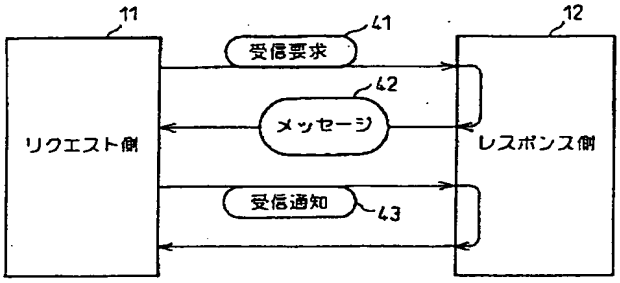
図７　ネームスペースを使ってメッセージＩＤを埋め込んだメッセージの例

```
<Order ssrs:messageId="12345678"
       xmlns="http://jp.fujitsu.com/order"
       xmlns:ssrs="http://jp.fujitsu.com/ssrs">
 <userInfo>
  :
 </userInfo>
</Order>
```

【図４】

図４

レスポンス側からリクエスト側にメッセージを送信する場合



【図８】

図８　任意のフィールドで一意性を保証するメッセージの例

```
<Order xmlns="http://jp.fujitsu.com/order">
 <orderId>order01234567</orderId>
 <userInfo>
  :
 </userInfo>
</Order>
```

【図５】

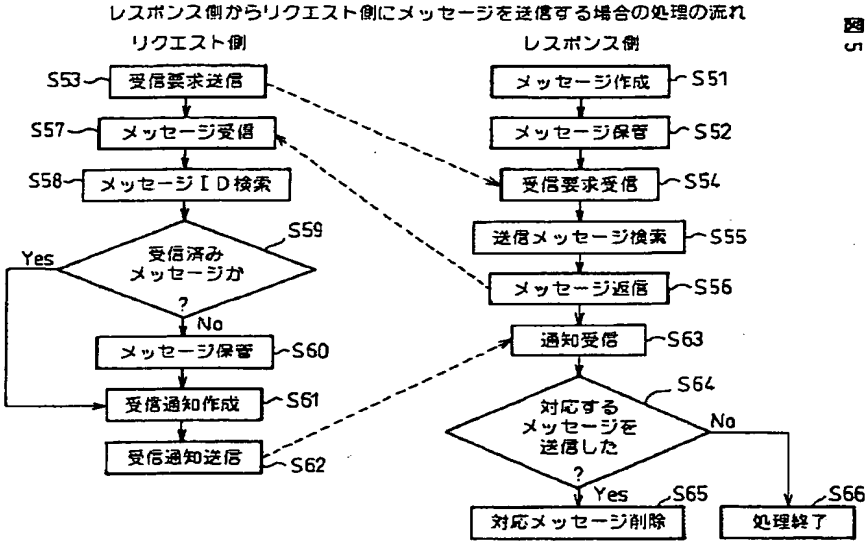


図５

【図９】

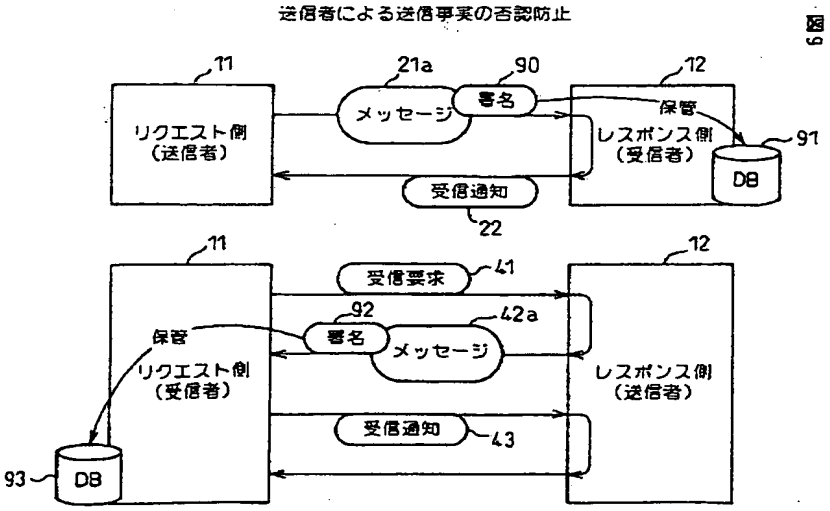
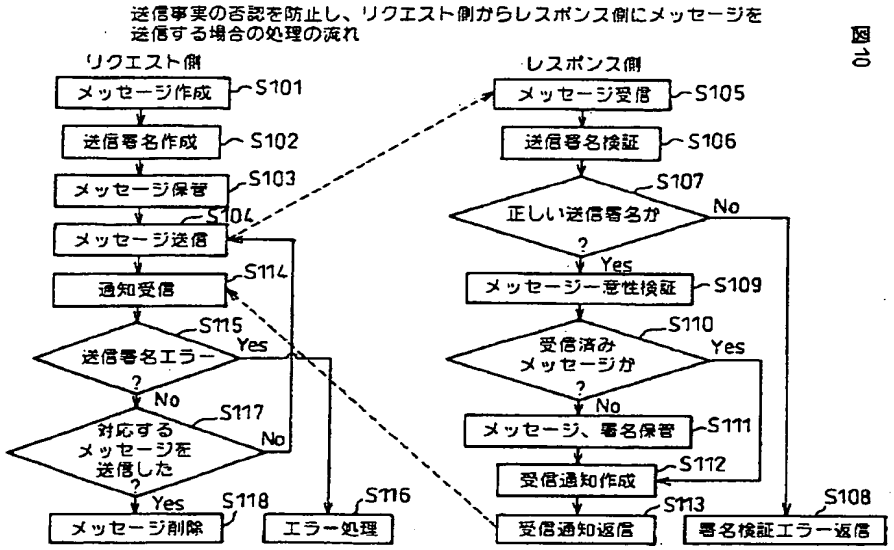
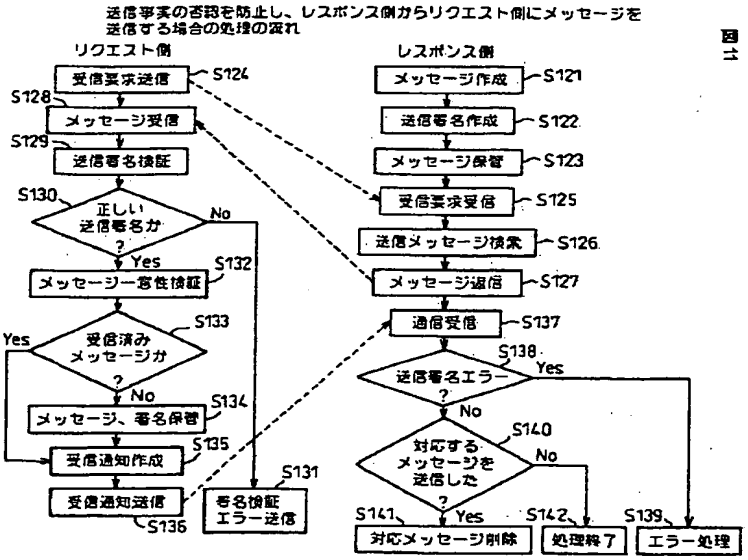


図９

【図１０】



【図１１】



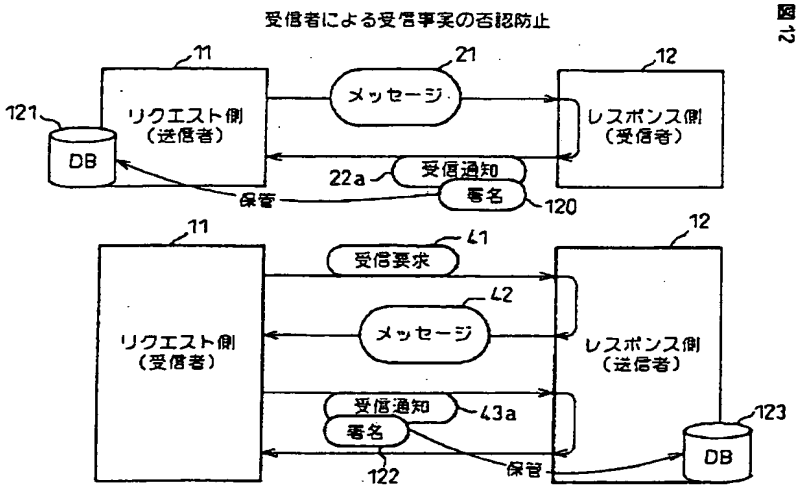
【図２５】

図 25

署名有り送信メッセージ（３）

```
    </ds:SignedInfo>
    <ds:SignatureValue>
     BL8jdfToEb1l/vXcMZNNjPOV...
    </ds:SignatureValue>
    <ds:KeyInfo>
     <wsse:SecurityTokenReference>
      <wsse:Reference URI="#X509Token"/>
     </wsse:SecurityTokenReference>
    </ds:KeyInfo>
   </ds:Signature>
  </wsse:Security>
 </SOAP:Header>
 <SOAP:Body>
    <gip:GetItemPrice xmlns:gip="Some-URI">
      <gip:itemnumber>product12345</gip:itemnumber>
    </gip:GetItemPrice>
 </SOAP:Body>
</SOAP:Envelope>
```

【図１２】


受信者による受信事実の否認防止
図12
11
21
12
121
リクエスト側
（送信者）
メッセージ
レスポンス側
（受信者）
DB
22a
受信通知
署名
120
保管
11
41
12
受信要求
42
リクエスト側
（受信者）
メッセージ
レスポンス側
（送信者）
受信通知
43a
署名
123
DB
122
保管

【図１３】

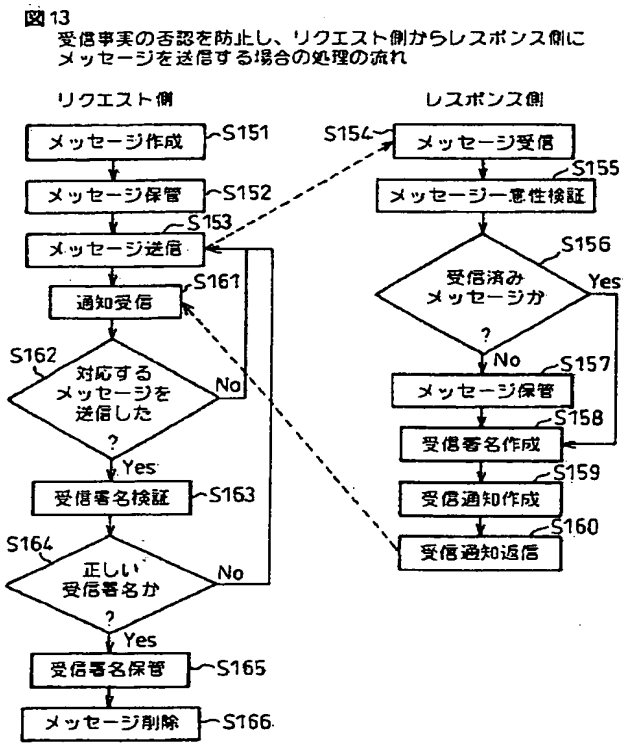


図13
受信事実の否認を防止し、リクエスト側からレスポンス側にメッセージを送信する場合の処理の流れ
リクエスト側
レスポンス側
メッセージ作成
S151
メッセージ保管
S152
メッセージ送信
S153
通知受信
S161
対応するメッセージを送信した
S162
No
Yes
受信署名検証
S163
正しい受信署名か
S164
No
Yes
受信署名保管
S165
メッセージ削除
S166
S154
メッセージ受信
S155
メッセージ一意性検証
受信済みメッセージか
S156
Yes
No
メッセージ保管
S157
受信署名作成
S158
受信通知作成
S159
受信通知返信
S160

【図１４】

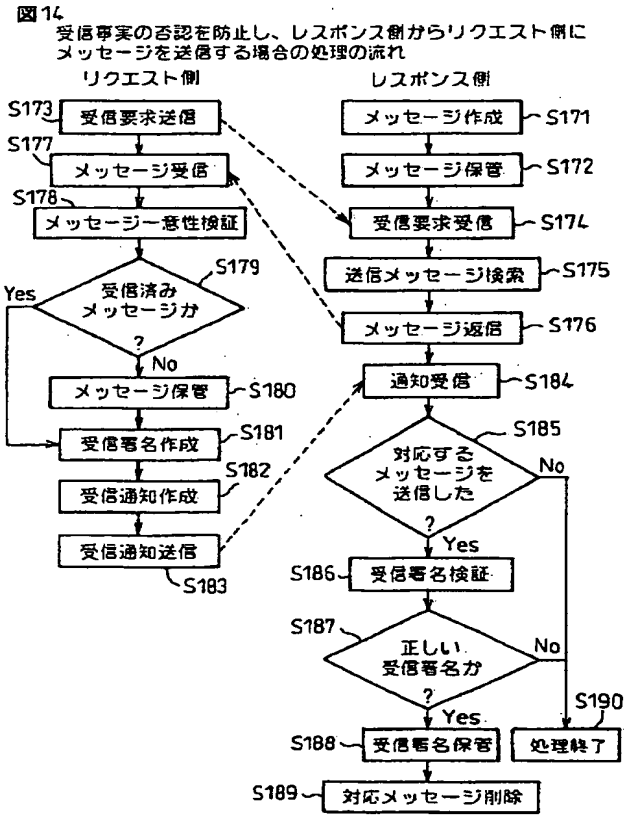


図14
受信事実の否認を防止し、レスポンス側からリクエスト側にメッセージを送信する場合の処理の流れ
リクエスト側
レスポンス側
S173
受信要求送信
S177
メッセージ受信
S178
メッセージ一意性検証
受信済みメッセージか
S179
Yes
No
メッセージ保管
S180
受信署名作成
S181
受信通知作成
S182
受信通知送信
S183
メッセージ作成
S171
メッセージ保管
S172
受信要求受信
S174
送信メッセージ検索
S175
メッセージ返信
S176
通知受信
S184
対応するメッセージを送信した
S185
No
Yes
S186
受信署名検証
S187
正しい受信署名か
No
Yes
S188
受信署名保管
S189
対応メッセージ削除
S190
処理終了

【図１５】

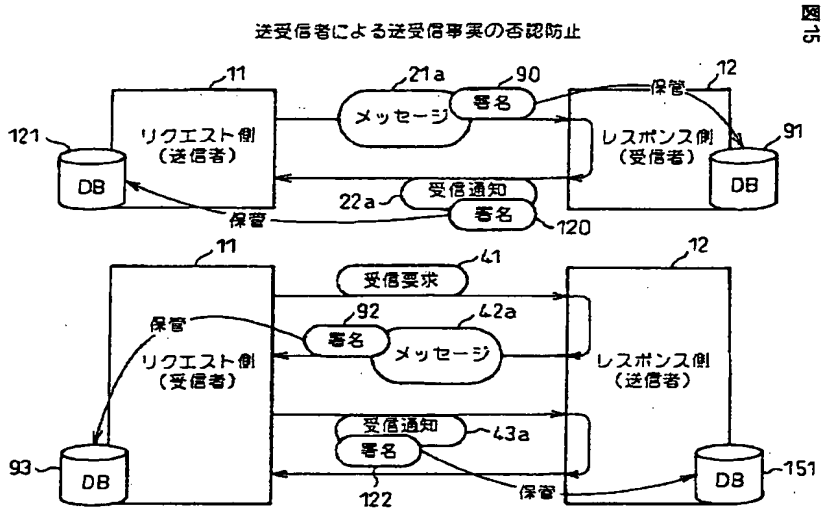


図15
送受信者による送受信事実の否認防止
11
21a
90
保管
12
メッセージ
署名
121
リクエスト側
（送信者）
レスポンス側
（受信者）
91
DB
DB
受信通知
22a
署名
保管
120
11
41
受信要求
12
92
42a
保管
署名
メッセージ
リクエスト側
（受信者）
レスポンス側
（送信者）
受信通知
43a
署名
93
DB
DB
151
122
保管

【図１６】

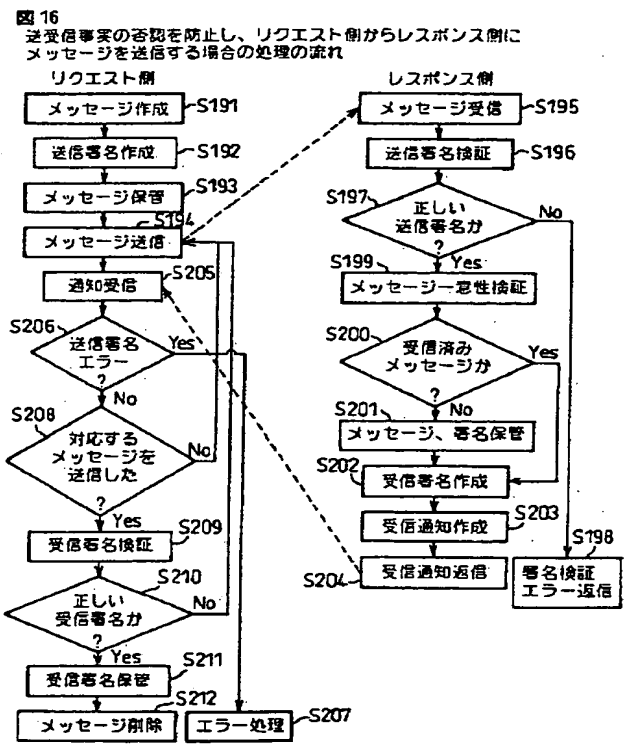


図16
送受信事実の否認を防止し、リクエスト側からレスポンス側に
メッセージを送信する場合の処理の流れ
リクエスト側
レスポンス側
メッセージ作成
S191
メッセージ受信
S195
送信署名作成
S192
送信署名検証
S196
メッセージ保管
S193
S197
正しい
送信署名か
No
S194
メッセージ送信
Yes
S199
メッセージ一意性検証
S205
通知受信
S200
受信済み
メッセージか
Yes
S206
送信署名
エラー
Yes
No
S201
No
メッセージ、署名保管
S208
対応する
メッセージを
送信した
No
S202
受信署名作成
Yes
S203
受信通知作成
S209
受信署名検証
S198
S210
正しい
受信署名か
No
S204
受信通知返信
署名検証
エラー返信
Yes
S211
受信署名保管
S212
メッセージ削除
エラー処理
S207

【図１７】

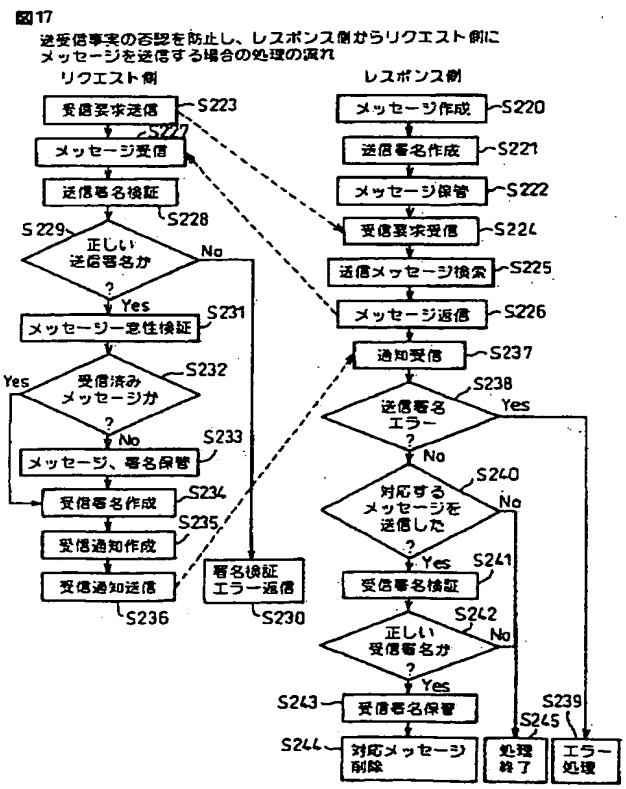
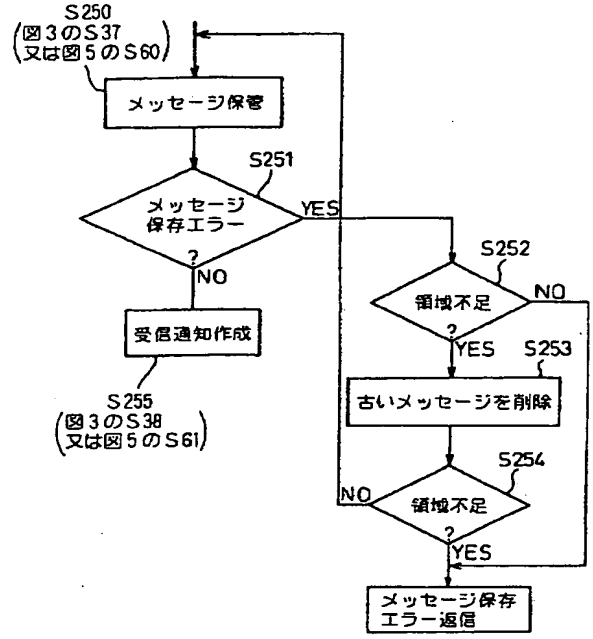


図17
送受信事実の否認を防止し、レスポンス側からリクエスト側に
メッセージを送信する場合の処理の流れ
リクエスト側
レスポンス側
受信要求送信
S223
メッセージ作成
S220
S227
メッセージ受信
送信署名作成
S221
送信署名検証
メッセージ保管
S222
S228
S229
正しい
送信署名か
No
受信要求受信
S224
送信メッセージ検索
S225
Yes
S231
メッセージ一意性検証
メッセージ返信
S226
S232
受信済み
メッセージか
Yes
通知受信
S237
Yes
No
S233
メッセージ、署名保管
S238
送信署名
エラー
Yes
No
S234
受信署名作成
S240
対応する
メッセージを
送信した
No
S235
受信通知作成
Yes
S241
受信署名検証
S236
受信通知送信
署名検証
エラー返信
S230
S242
正しい
受信署名か
No
Yes
S243
受信署名保管
S239
S245
S244
対応メッセージ
削除
処理
終了
エラー
処理

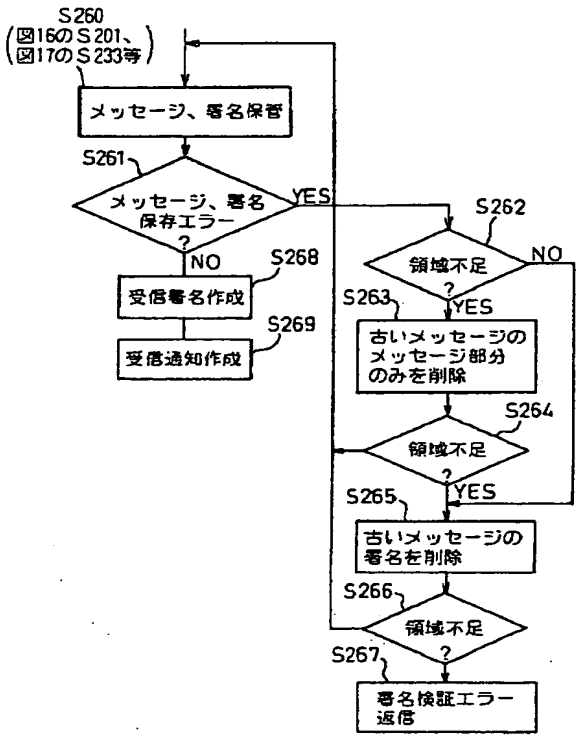
【図１８】

図18



【図１９】

図19



【図２０】

図20

メッセージの送信要求

```
<SOAP:Envelope
 xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP:Header>
  <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
   SOAP:mustUnderstand="1">
      <rm:From>requester@anyuri.com</rm:From>
      <rm:To>responder@someuri.com</rm:To>
      <rm:Service>urn:services:ItemQuoteService</rm:Service>
      <rm:MessageType>Request</rm:MessageType>
  </rm:MessageHeader>
 </SOAP:Header>
 <SOAP:Body>
 </SOAP:Body>
</SOAP:Envelope>
```

【図２８】

図28

署名有り受信確認メッセージ（３）

```
        </ds:Transforms>
        <ds:DigestMethod Algorithm=
            "http://www.w3.org/2000/09/xmldsig#sha1"/>
        <ds:DigestValue>EULddytSo1...</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>
      BL8jdfToEb1l/vXcMZNNjPOV...
    </ds:SignatureValue>
    <ds:KeyInfo>
      <wsse:SecurityTokenReference>
        <wsse:Reference URI="#X509Token"/>
      </wsse:SecurityTokenReference>
    </ds:KeyInfo>
  </ds:Signature>
 </wsse:Security>
</SOAP:Header>
<SOAP:Body>
</SOAP:Body>
</SOAP:Envelope>
```

【図２１】

署名無し送信メッセージ　図21

```
<SOAP:Envelope
 xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP:Header>
   <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
    SOAP:mustUnderstand="1">
       <rm:From>requester@anyuri.com</rm:From>
       <rm:To>responder@someuri.com</rm:To>
       <rm:Service>urn:services:ItemQuoteService</rm:Service>
       <rm:MessageType>Message</rm:MessageType>
       <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
       <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
   </rm:MessageHeader>
   <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
    SOAP:mustUnderstand="1">
     <rm:AckRequested SOAP:mustunderstand="1" rm:version="1.0"
rm:signed="false" />
       <rm:DuplicateElimination/>
   </rm:ReliableMessage>
 </SOAP:Header>
 <SOAP:Body>
     <gip:GetItemPrice xmlns:gip="Some-URI">
       <gip:itemnumber>product12345</gip:itemnumber>
     </gip:GetItemPrice>
 </SOAP:Body>
</SOAP:Envelope>
```

【図２２】

署名無し受信確認メッセージ　図22

```
<SOAP:Envelope
 xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP:Header>
   <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
    SOAP:mustUnderstand="1">
       <rm:From>responder@someuri.com</rm:From>
       <rm:To>requester@anyuri.com</rm:To>
       <rm:Service>urn:services:ItemFilingService</rm:Service>
       <rm:MessageType>Acknowledgment</rm:MessageType>
       <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
       <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
       <rm:RefToMessageId>20020907-44-070@uri.com</rm:RefToMessageId>
   </rm:MessageHeader>
   <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
    SOAP:mustUnderstand="1">
       <rm:Acknowledgment SOAP:mustunderstand="1" rm:version="1.0" />
   </rm:ReliableMessage>
 </SOAP:Header>
 <SOAP:Body>
 </SOAP:Body>
</SOAP:Envelope>
```

【図２３】

署名有り送信メッセージ（１）

図23

```
<SOAP:Envelope
 xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP:Header>
  <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
   SOAP:mustUnderstand="1">
     <rm:From>requester@anyuri.com</rm:From>
     <rm:To>responder@someuri.com</rm:To>
     <rm:Service>urn:services:ItemQuoteService</rm:Service>
     <rm:MessageType>Message</rm:MessageType>
     <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
     <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
  </rm:MessageHeader>
  <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
   SOAP:mustUnderstand="1">
    <rm:AckRequested SOAP:mustunderstand="1" rm:version="1.0"
rm:signed="false"/>
     <rm:DuplicateElimination/>
  </rm:ReliableMessage>
  <wsse:Security xmlns:wsse=" http://schemas.xmlsoap.org/ws/2002/04/secext">
   <wsse:BinarySecurityToken
    ValueType="wsse:X509v3"
    EncodingType="wsse:Base64Binary"
    Id="X509Token">
    MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
```

【図２４】

署名有り送信メッセージ（２）

図24

```
    </wsse:BinarySecurityToken>
    <ds:Signature xmlns:ds=" http://www.w3.org/2000/09/xmldsig#">
     <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <ds:SignatureMethod Algorithm=
          "http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <ds:Reference URI="">
       <ds:Transforms>
        <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
         <ds:XPath> ancestor-or-self::rm:MessageHeader or ancestor-or-self::rm:Reliable
Message or ancestor::SOAP:Body</ds:XPath>
        </ds:Transform>
        <ds:Transform Algorithm=
            "http://www.w3.org/2001/10/xml-exc-c14n#"/>
       </ds:Transforms>
       <ds:DigestMethod Algorithm=
           "http://www.w3.org/2000/09/xmldsig#sha1"/>
       <ds:DigestValue>EULddytSo1...</ds:DigestValue>
      </ds:Reference>
```

【図２６】

署名有り受信確認メッセージ（１）　図26

```
<SOAP:Envelope
 xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP:Header>
  <rm:MessageHeader xmlns:rm="http://schemas.xmlsoap.org/rm"
   SOAP:mustUnderstand="1">
     <rm:From>responder@someuri.com</rm:From>
     <rm:To>requester@anyuri.com</rm:To>
     <rm:Service>urn:services:ItemFilingService</rm:Service>
     <rm:MessageType>Acknowledgment</rm:MessageType>
     <rm:MessageId>20020907-045261-0450@anyuri.com</rm:MessageId>
     <rm:Timestamp>2002-09-07T10:19:07</rm:Timestamp>
     <rm:RefToMessageId>20020907-44-070@uri.com</rm:RefToMessageId>
  </rm:MessageHeader>
  <rm:ReliableMessage xmlns:rm="http://schemas.xmlsoap.org/rm"
   SOAP:mustUnderstand="1">
     <rm:Acknowledgment SOAP:mustunderstand="1" rm:version="1.0" />
  </rm:ReliableMessage>
  <wsse:Security xmlns:wsse=" http://schemas.xmlsoap.org/ws/2002/04/secext">
   <wsse:BinarySecurityToken
     ValueType="wsse:X509v3"
     EncodingType="wsse:Base64Binary"
     Id="X509Token">
```

【図２７】

署名有り受信確認メッセージ（２）　図27

```
     MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
   </wsse:BinarySecurityToken>
   <ds:Signature xmlns:ds=" http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
     <ds:CanonicalizationMethod Algorithm=
         "http://www.w3.org/2001/10/xml-exc-c14n#" />
     <ds:SignatureMethod Algorithm=
         "http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
     <ds:Reference URI="">
      <ds:Transforms>
       <ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
         <ds:XPath> ancestor-or-self::rm:MessageHeader or ancestor-or-self::rm:Reliable
Message </ds:XPath>
       </ds:Transform>
       <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#" />
      </ds:Transforms>
      <ds:DigestMethod Algorithm=
          "http://www.w3.org/2000/09/xmldsig#sha1" />
      <ds:DigestValue>EULddytSo1...</ds:DigestValue>
     </ds:Reference>
     <ds:Reference URI="rm:20020907-045261-0450@anyuri.com">
     <ds:Transforms>
       <ds:Transform Algorithm=
          "http://www.w3.org/2001/10/xml-exc-c14n#" />
```

フロントページの続き

Ｆターム(参考)　5K018 BA03 FA02
5K034 AA06 DD01 EE10 HH01 HH02
HH11 MM05